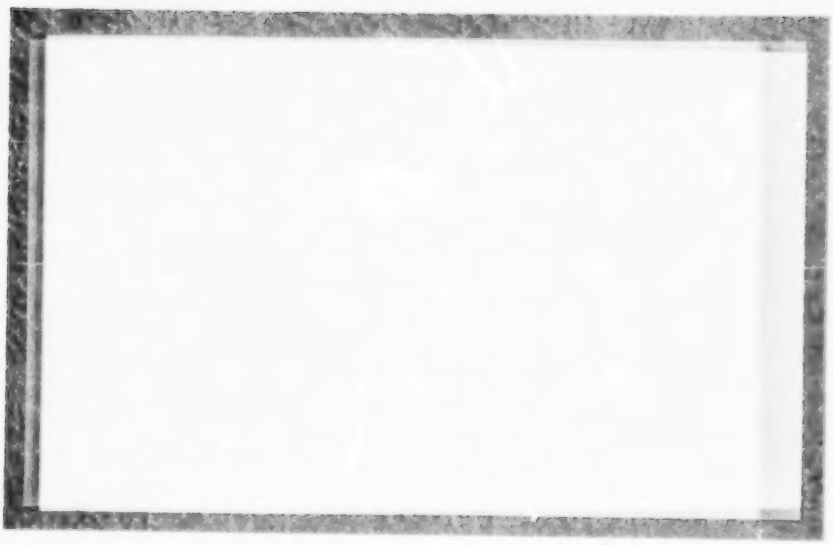
UNIVERSITÉ DE MONTRÉAL



CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

AN ALGORITHM FOR THE SPATIAL PRICE EQUILIBRIUM PROBLEM ON A GENERAL NETWORK IN THE SPACE OF THE PATH FLOWS

by

Omar Drissi-Kaïtouni
and
Michael Florian

This research was supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada.

Cette étude a été publiée grâce à une subvention du fonds F.C.A.R. pour l'aide et le soutien à la recherche.

Université de Montréal
Centre de recherche sur les transports - Publication #517
Avril 1987

Dépôt légal - 2e trimestre 1987

ABSTRACT

The Spatial Price Equilibrium on a General Network may be formulated as a nonlinear cost mathematical programming problem with simple constraints, when the decision variables are the path flows. The solution of this problem is difficult 'a priori' due to the very large number of variables (paths) and the impracticality of generating all the paths from all the origins to all the destinations.

In this paper we develop a Gauss-Seidel-Newton-Projection algorithm and combine with a Restriction strategy. That makes it unnecessary to generate 'a priori' all the paths. This algorithm may be further improved by exploiting the equivalence between the Spatial Price Equilibrium on a General Network and the Network Equilibrium.

The algorithms have been implemented on the Cyber 835 of the Université de Montréal. Computational results that we present in this paper demonstrate the efficiency of the proposed solution algorithms.

Key words : Spatial Price Equilibrium on a General Network, Network Equilibrium, Gauss-Seidel, Newton, Restriction.

RÉSUMÉ

Le problème d'équilibre spatial sur réseau général peut être formulé comme un problème de programmation mathématique non linéaire avec contraintes de non négativité seulement, lorsque les variables considérées pour la résolution du problème sont les flots sur les chemins. La résolution de ce problème est difficile 'a priori' à cause du nombre très élevé de variables (chemins) et de la difficulté à générer tous les chemins de toutes les origines à toutes les destinations.

Dans cet article, nous développons un algorithme de type Gauss-Seidel-Newton-Projection auquel nous associons une stratégie de restriction, de façon à ce qu'il ne soit pas nécessaire de générer a priori tous les chemins. Cet algorithme est ensuite amélioré à l'aide de l'équivalence entre le problème d'équilibre spatial sur réseau général et le problème d'équilibre de réseau.

Les algorithmes proposés ont été implantés sur le CYBER 835 de l'Université de Montréal via un programme FORTRAN. Les résultats numériques, que nous présentons dans cet article, démontrent l'efficacité de ces algorithmes.

Mots-clés : équilibre spatial sur réseau général, équilibre de réseau, Gauss-Seidel, Newton, restriction.

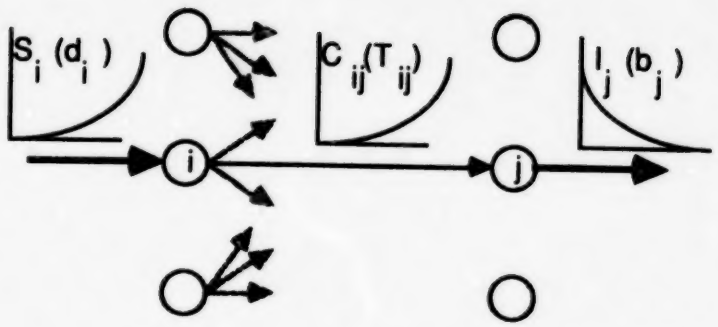
INTRODUCTION

The solution of the classical Spatial Price Equilibrium Problem (Samuelson, 1952; Takayama and Judge, 1971) determines, on a bipartite transportation network, the quantities of goods sold, bought and transported from origins to destinations where a descriptive price equilibrium is achieved.

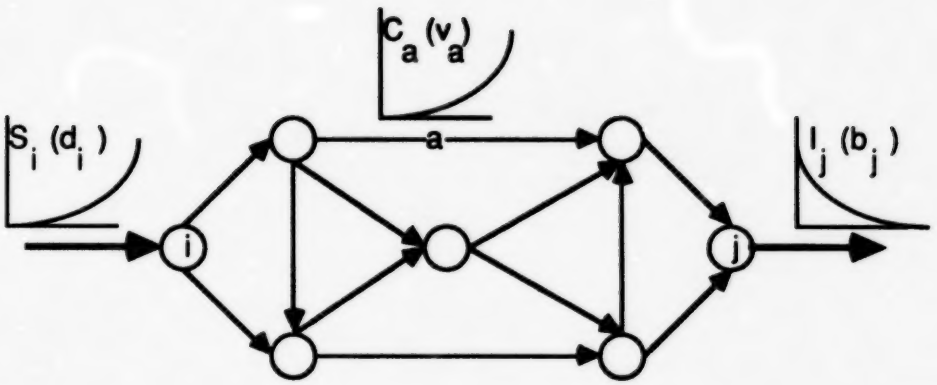
At each origin (destination), the supply (demand) price is modeled by a monotone (monotone decreasing) supply (demand) function. The transportation costs may be linear or monotone functions of the quantities transported. Such a problem is illustrated in Figure 1-a. Due to its similarity to the Hitchcock (1941) transportation problem, when the quantities offered and bought are constant, this problem is also known as Generalized Transportation Problem (McKinnon, 1975; Florian, 1981).

A more general version of this problem, illustrated in Figure 1-b, is where the Spatial Price Equilibrium is established over a general network that offers competing paths for the transportation of goods between origins and destinations. This variant of the problem on a general network is sometimes referred to as the General Spatial Price Equilibrium Problem. In the following we shall refer to it as the GSPE problem.

In a latter section we will undertake the mathematical formulation of the GSPE problem. It has been shown that this problem may be formulated as a nonlinear cost mathematical program with linear constraints. Florian and Los (1982) formulated the problem in the space of the path flows, while Tobin and Friesz (1983) formulated the problem in the space of the link flows. The latter model has the advantage of requiring at most $|A| + 2|N|$ variables, where $|A|$ is the number of arcs and $|N|$ is the number of nodes in the network.



- a -



- b -

Figure 1-a, 1-b

The formulation in the space of path flow requires 'a priori' a very large number of variables, since the number of paths in a general network grows exponentially with the size of the network. Its solution provides more information, since in addition to the quantities supplied and bought, one obtains the routing of the goods from origins to destinations; this information is not available when solving the problem in the space of the arc flows.

The literature is not replete with solution approaches for the GSPE problem. Florian and Los (1982) suggested the adaptation of the linear approximation method for the problem formulated in the space of path flows; they did not test the algorithm and noted that it is but the simplest of other algorithms which may be applied. Tobin and Friesz (1983) applied the reduced gradient algorithm implemented in MINOS (Murtagh and Saunders, 1977) to the problem formulated in the space of arc flows.

In this paper, we develop and test the efficiency of a Gauss-Seidel-Newton Projection algorithm combined with a restrictive policy which we denote GSNPR. This algorithm does not require the 'a priori' generation of all the paths in the network; the necessary paths are generated as required by shortest path computations.

The paper is organized as follows : first we recall the Gauss-Seidel type algorithm for minimizing nonlinear cost functions subject to simple constraints; then we consider Newton methods for the solution of the subproblems of the Gauss-Seidel approach; in the third section, a GSNPR algorithm is developed for the GSPE problem; implementation details are discussed in the fourth section and the last section contains the numerical results obtained.

1. GAUSS-SEIDEL TYPE ALGORITHMS

In this section, we recall the definition of a Gauss-Seidel type algorithm for non linear programs with simple constrains, and state a convergence theorem.

Consider the program : (P) $\begin{cases} \min \ f(x) \\ \text{s.t.} \ \ x \in X = \{x \in E^n \mid x \geq 0\} \end{cases}$

Let $f_i^k(x_i) = f(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i, x_{i+1}^k, \ldots, x_n^k)$

and $g_i^k(x_i) = g(x_1^{k+1}, \ldots, x_{i-1}^{k+1}, x_i, x_{i+1}^k, \ldots, x_n^k)$,

where g_i is the partial derivative of f relatively to x_i.

The Gauss-Seidel algorithm that we consider in order to solve (P) is the following :

(GS) : 0- Initialize x^0, k=0 .

1- i=1 TO n DO.

Solve the 1-variable subproblem in x_i :

(SP_i^k) : min $f_i^k(x_i)$ s.t. $x_i \geq 0$

and let x_i^{k+1} be the optimal solution to SP_i^k .

2- Optimality test : (STOP) OR (k=k+1 and RETURN to 1-) .

THEOREM 1 : If f is a continuous convex function, bounded from below and attains its minimum on X, then any sequence generated by (GS) converges to the optimal solution x* of (P).

We do not give the proof of this theorem, since this result is classical (see Luenberger, 1973, p. 227-230; Ortega and Rheinboldt, 1970, p. 222-226; ...). On the other hand we present, in a later section of this paper, the convergence proof for a similar algorithm applied to the GSPE problem.

It is evident that the most important part of the algorithm (GS) described above is the solution of the subproblem SP_i^k. The next section deals with Newton methods to solve the 1-variable subproblems SP_i^k, when $f \in C^2$.

2. SOLUTION OF THE SUBPROBLEM

In order to simplify the presentation, we consider a single subproblem and omit the indexes i and k; we have to solve : (SP) : min $f(x)$ s.t. $x \geq 0$ where $x \in E^1$, f is convex and twice differentiable. $g = f'$ is the first derivative of f, and $g' = f''$ is its second derivative.

This section is divided into two parts. In the first, we recall the "Restricted Step Method" described in Fletcher (1980, Section 5). This method has the advantage of the convergence to a Kuhn-Tucker point (if any) even if the Hessian matrix (here f") is not positive definite everywhere. This method maintains the quadratic convergence rate but requires more function evaluations at each iteration than the classical Newton method, which we describe in the second part of this section.

Restricted Step Method (Nr) :

The restricted step method (Fletcher, 1980) applied to the problem : min $f(x)$ where x is in E^1, can be stated as follows :

(Nr) : i) given x^k and h^k, compute $g(x^k)$ and $g'(x^k)$

ii) let d^k be the optimal solution to the problem :

$$\min q^k(d) \text{ s.t. } d \leq h^k$$

$$\text{where } q^k(d) = f(x^k) + g(x^k)d + g'(x^k)d^2/2$$

iii) evaluate $r^k = [f(x^k) - f(x^k + d^k)]/[f(x^k) - q^k(d^k)]$

iv) if $r^k < 0{,}25$ then let $h^{k+1} = d^k/4$

if $r^k > 0{,}75$ and $|d^k| = h^k$ then $h^{k+1} = 2h^k$

else $h^{k+1} = h^k$

v) if $r^k < 0$ then $x^{k+1} = x^k$

else $x^{k+1} = x^k + d^k$

where h^k is a radius initially arbitrarily chosen to restrict the magnitude of the descent direction.

The constants 0,25, 0,75, 2 and 4 are chosen arbitrarily, but the algorithm is not very sensitive to their variations (Fletcher, 1980).

THEOREM 2 : For this algorithm, if $x^k \in B \subset E^n$ for all k, where B is bounded, and if $f \in C^2$ and $\|g'\| < G$ on B, then there exists an accumulation point x^∞ which satisfies the first and second order necessary conditions.

Proof : See Fletcher, 1980.

We must note that this algorithm solves the problem : min f(x). To solve (SP) : min f(x) s.t. $x \geq 0$, the only modification to Nr must be to replace $x^k + d^k$ in -v- by max $(0, x^k + d^k)$, the projection on the positive orthant.

For a convex function f that attains its minimum of X, the accumulation point of the theorem is not only a local but a global minimum due to the convexity of f.

We recall that Nr maintains the quadratic convergence rate of the Newton method (Fletcher, 1980), but it is clear that at each iteration, it is necessary to carry out more function evaluations than in the classical Newton method (computation of $f(x^k)$ and evaluation of r^k in -iii-, and the tests in -iv- and -v-).

In order to use the classical Newton method to solve (SP), and to be insured that it converges globally, some mild additional conditions are to be made on the function g.

Classical Newton Method with Projection (NP)

We first present the method (NP) to solve (SP), and then state a convergence theorem. Recall that (SP) is :

$$(SP) : \min f(x) \text{ s.t. } x \geq 0 , \ x \in E^1$$

(NP) : i) Initialize $x^0 \geq 0$ and $k=0$;

ii) IF $x^k g(x^k) = 0$ and $g(x^k) > 0$ Go to -iv- ;

Else let $d^k = -g(x^k)/g'(x^k)$ and

$x^{k+1} = \max (0, x^k + d^k)$;

iii) $k=k+1$ and Goto -ii- ;

iv) Stop .

THEOREM 3 : If $f \in C^2$ is convex and attains its minimum on $[0,\infty)$, if $g=f'$ satisfies one of the three following conditions :

i) g is convex on $[0,\infty)$

ii) g is concave on $[0,\infty)$

iii) g is concave on $[0,p]$ and convex on $[p,\infty)$ for a given $p \geq 0$

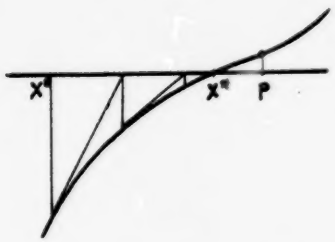
Then any sequence generated by (NP) converges to the optimal solution of (SP), and for a certain K, the convergence is quadratic for $k \geq K$.

Proof : Since cases i and ii are particular cases of case iii ($p=0$ for case i and $p=\infty$ for case ii), we will make the proof for case iii only.

For case iii, we will make the proof in two steps. In the first one, we will relax the constraint $x \geq 0$. We will complete the proof in the second step by adding these constraints $x \geq 0$.

Step 1 : Six cases are possible, and we will prove convergence for each of these cases :

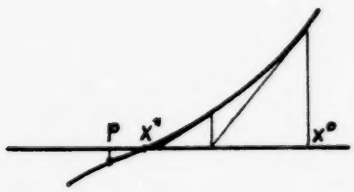
Case 1 : $x^0 < x^* < p$



-g is convex and monotone on $(-\infty, p]$; the proof is in Ortega and Rheinboldt (1970), and :

$$x^k \to x^* \text{ for } k \geq 0$$

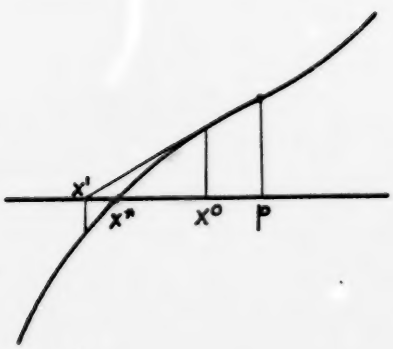
Case 2 : $p < x^* < x^0$



g is convex and monotone on $(p, \infty]$; the proof is in Ortega and Rheinboldt (1970), and :

$$x^k \to x^* \text{ for } k \geq 0$$

Case 3 : $x^* < x^0 < p$

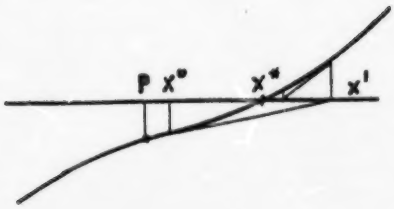


x^1 is obtained from :

$g(x^0)+g'(x^0)(x^1 - x^0) = 0$ but $0=g(x^*)$ then $0=g(x^*)<g(x^0)+g'(x^0)(x^*-x^0)$ then $g'(x^0)(x^1-x^0) < g'(x^0)(x^*-x^0)$. Since $g'(x^0) > 0$ (if not x^0 would be optimal) then $x^1 < x^*$ and we fall in case 1, and

$$x^k \to x^* \text{ for } k > 1$$

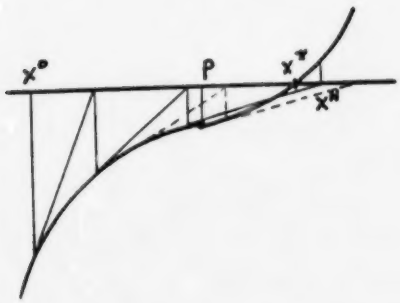
Case 4 : $p<x^0<x^*$



The proof is similar to the one of case 3, and convexity of g on $[p,x^*]$ will let $x^1>x^*$ and we fall in case 2, and

$$x^k \downarrow x^* \text{ for } k > 1$$

Case 5 : $x^0<p<x^*$



x^0 is in the concave part; let k=0 and suppose x^k in the concave part. If $x^{k+1}>p$ we fall in case 2 or 4 (proved). If not x^{k+1} is also in the concave part. Suppose that for any k we stay in the concave part, then knowing that f is monotone and that $g(x^*)=0$, then $g(x^k)$ $g(x^*)=0$ for any k since $x^k<p<x^*$, then $x^{k+1}-x^k = -g(x^k)/g'(x^k) > 0$ because $g'(x^k)>0$. That is true because g is monotone then $g'(x^k)>0$, and if it was zero then x^k would have been optimal which is in contradiction with $p<x^*$. The sequence $\{x^k\}$ is monotone and bounded (by p), then it converges. At the limit ($\lim_{k\to\infty}(x^{k+1}-x^k)=0=\lim_{k\to\infty}(g(x^k)/ g'(x^k))\to g(x)=0$, that is inconsistent with the fact that the solution is in the convex part. One can then find an integer N such that $x^N>x^*>p$ and then $x^k \uparrow x^N$ for k=0 .. N-1 and $x^k \downarrow x^*$ for k=N .. ∞ .

Case 6 : $x^*<p<x^0$

The proof is similar to the one of the previous case an :

$$x^k \downarrow x^N \text{ for } k=0 \text{ .. } N-1 \text{ and}$$

$$x^k \uparrow x^* \text{ for } k=N \text{ .. } \infty$$

Step 2 : If g is constrained on $[0,\infty)$ by $x \geq 0$, and that $x^0 \geq 0$, there are two cases that may happen :

1- $x^* > 0$

The only cases where $x^k + d^k < 0$ may happen are the cases 3 and 6 with j=0 for case 3 and j=N-1 for case 6. The algorithm (NP) will then take $x^{j+1} = 0$ and since $x^k \uparrow x^*$ for $k \geq j$ then there will be convergence.

2- $x^* < 0$

Recall that g is monotone then $g(0) \geq 0$. The algorithm will stop if $x^k + d^k < 0$ with 0 as a solution and this solution is optimal since $0 g(0) = 0$ and $g(0) \geq 0$. o

The Newton method is known to converge for cases i- and ii- which are special cases of iii-. Case iii- is pertinent for spatial price equilibrium problems where the functions g, undefined as yet, often have the same behaviour as in iii-.

In the following, we refer to

GSNP : GS algorithm where the subproblems are solved by the NP algorithm, and

GSNr : GS algorithm where the subproblems are solved by the Nr algorithm.

In the next section, we adapt the algorithms described here to the GSPE problem. We exploit some properties of flows in networks in order to improve the efficiency of the algorithms GSNP and GSNr for this particular problem.

3. THE GSPE PROBLEM

We consider the set $W=\{(i,j),\ i \in O,\ j \in D\}$ of origin-destination pairs. A commodity is produced at origins i and consumed at destinations j.

In the model first proposed by Samuelson (1952), and Takayama and Judge (1971), on a bipartite network with constant transportation costs C_{ij}, $(i,j) \in W$, the equilibrium conditions are :

$$S_i(d_i) + C_{ij} - I_j(b_j) \begin{cases} = 0 & \text{if } T_{ij} > 0 \\ > 0 & \text{if } T_{ij} = 0 \end{cases} \qquad (1)$$

where $S_i(d_i)$ is the price per unit for supplying quantity d_i and $I_j(b_j)$ is the buying price per unit for quantity b_j.

The variables d_i, T_{ij} and b_j satisfy flow conservation and non-negativity constraints :

$$d_i = \sum_j T_{ij}, \text{ for all } i \; ; \quad b_j = \sum_i T_{ij}, \text{ for all } j \; ; \qquad (2)$$

$$T_{ij} > 0, \text{ for all } (i,j) \; . \qquad (3)$$

It is well known that this problem has the equivalent minimization problem.

$$\text{Min } f(T) = \sum_i \int_0^{d_i} S_i(x)dx + \sum_i \sum_j C_{ij}T_{ij} - \sum_j \int_0^{b_j} I_j(z)dz \qquad (4)$$

subject to (2) and (3)

When the transportation costs are flow dependent, the equilibrium conditions change slightly to :

$$S_i(d_i) + C_{ij}(T_{ij}) - I_j(b_j) \begin{cases} = 0 & \text{if } T_{ij} > 0 \\ > 0 & \text{if } T_{ij} = 0 \end{cases} \qquad (5)$$

and this problem, known as the generalized transportation problem, after replacing for d_i and b_j by using (2), has the equivalent minimization problem (Florian and Los, 1982) :

$$\text{Min } f(T) = \sum_i \int_0^{\sum_j T_{ij}} S_i(x)dx + \sum_i \sum_j \int_0^{T_{ij}} C_{ij}(y)dy - \sum_j \int^{\sum_i T_{ij}} I_j z(dz) \qquad (6)$$

subject to $T_{ij} > 0$, for all (i,j).

This convex program has the form : min f(T), subject to T>0, it is clear that algorithms GSNP and GSNr may be naturally adapted to solve (6).

The formulation of the GSPE problem considers a general network. Origins and destinations are joined by paths. Consequently, the equilibrium conditions may be written (Florian and Los, 1982) :

$$S_i(d_i) + \sum_k \delta_{ak} C_a(v_a) - I_j(b_j) \begin{cases} = 0 & \text{if } h_k > 0,\ k \in K_{ij}, \quad \text{for all } (i,j) \\ > 0 & \text{if } h_k = 0,\ k \in K_{ij}, \quad \text{for all } (i,j) \end{cases} \tag{7}$$

where K_{ij} is the set of paths joining origin i and destination j and

$$\delta_{ak} = \begin{cases} 1 & \text{if arc } a \in k \\ 0 & \text{otherwise .} \end{cases}$$

Consequently the arc flow $v_a = \sum_{i,j} \sum_{k \in K_{ij}} \delta_{ak} h_k$.

The problem (7) has a convex cost equivalent minimization problem (Florian and Los, 1982) which may be stated as :

$$\text{Min } f(h) = \sum_i \int_0^{\sum_j \sum_{k \in K_{ij}} h_k} S_i(x)dx + \sum_a \int_0^{\sum_{i,j} \sum_{k \in K_{ij}} \delta_{ak} h_k} C_a(y)dy - \sum_j \int_0^{\sum_i \sum_{k \in K_{ij}} h_k} I_j(z)dz$$

$$\text{subject to } h_k > 0,\ k \in K_{ij} \text{ for all } (i,j) \tag{8}$$

This problem is also of the form : min f(h), subject to h>0, so it is possible to use the previous algorithms to solve it. A drawback of this formulation is that the number of paths grows exponentially with the size of the network and in general large problems would be difficult to solve with those algorithms.

In the next section, we will exploit the network structure to reduce the number of variables needed to solve the problem, but before doing so, we juxtapose the GSPE problem and the Network Equilibrium problem. It is

helpful to pose the GSPE problem as a Network Equilibrium problem (see also Dafermos, 1986).

In order to do so the network is augmented with an additional node m and arcs (m,i), for all i and (j,m), for all j. The costs and flows of these arcs are respectively:

$$C_{mi} = S_i \quad \text{and} \quad v_{mi} = d_i, \qquad \text{for all } i$$

$$C_{jm} = -I_j \quad \text{and} \quad v_{jm} = b_j, \qquad \text{for all } j$$

On this expanded network, the equilibrium conditions (7) become :

$$C_{mi}(v_{mi}) + \sum_a \delta_{ak}\, C_a(v_a) + C_{jm}(v_{jm}) \begin{cases} = 0 & \text{if } h_k > 0 \\ > 0 & \text{if } h_k = 0 \end{cases} \quad k \in K_{ij}, \text{ for all } (i,j) \qquad (9)$$

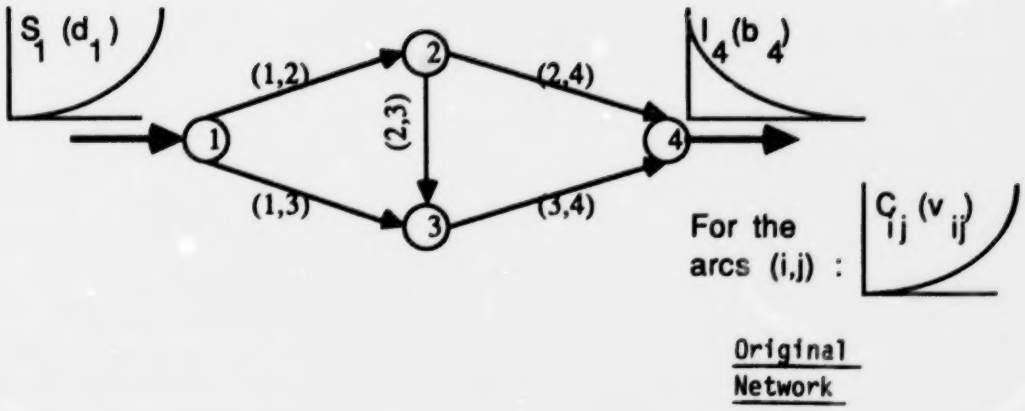
By considering the flows on arcs (m,i) and (j,m) as regular arcs of the expanded network, (9) may be rewritten simply as

$$\sum_a \delta_{ak}\, C_a(v_a) \begin{cases} = 0 & \text{if } h_k > 0 \\ > 0 & \text{if } h_k = 0 \end{cases} \quad k \in K_{ij}, \text{ for all } (i,j) \qquad (10)$$

which are in the same form as the equilibrium conditions for the Network Equilibrium problem. A path $k \in K_{ij}$ begins at m, terminates at m and passes through nodes i and j.

The conditions (10) correspond to the equilibrium conditions of the Network Equilibrium Problem on the augmented network. The Spatial Price Equilibrium Problem can then be seen as a Network Equilibrium Problem with one origin-destination pair (m,m).

The arcs a are this time in the augmented set A^+, the set of original arcs A augmented by the new arcs (m,i)'s and (j,m)'s, and a path k must now be seen as the path k from the origin m to the destination m, passing through the nodes i and j. Figure 2 illustrates the passage from problem (7) to problem (10) . On the augmented network, one must note that there are negative costs on some arcs (the (j,m)'s arcs):


$S_1(d_1)$
$l_4(b_4)$
(1,2)
(2,3)
(2,4)
(1,3)
(3,4)
1
2
3
4
For the arcs (i,j) :
$C_{ij}(v_{ij})$
Original Network

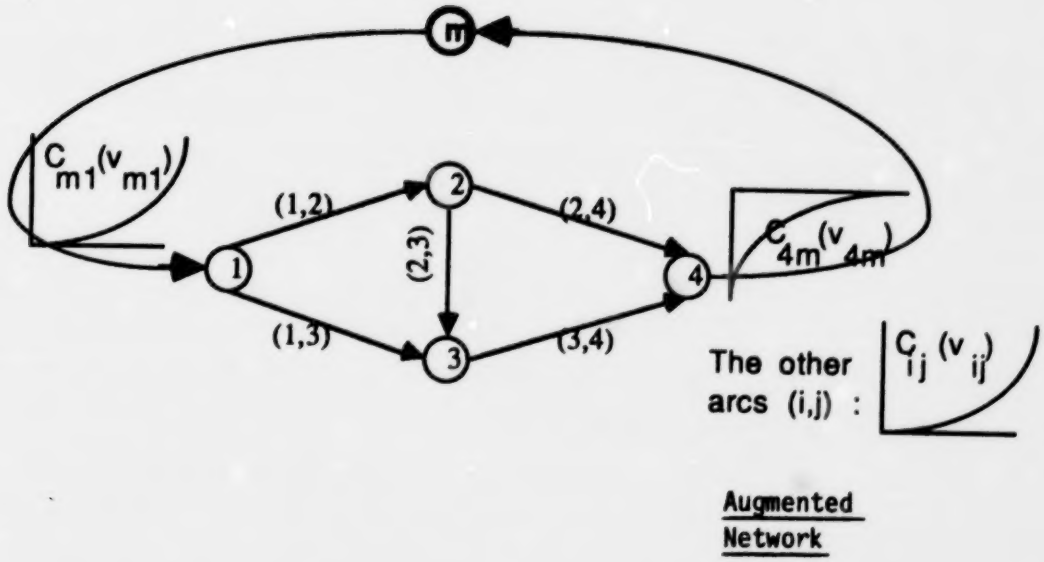

m
$C_{m1}(v_{m1})$
(1,2)
(2,3)
(2,4)
(1,3)
(3,4)
$C_{4m}(v_{4m})$
1
2
3
4
The other arcs (i,j) :
$C_{ij}(v_{ij})$
Augmented Network

Figure 2

Solution of the Spatial Price Equilibrium Problem - GSPE

First, we return to the problem formulated on the original network and apply to it a GSN type algorithm. Then, we exploit the representation in the augmented network. Before doing so we state the following theorem that will be useful in reducing the number of variables (paths) :

THEOREM 4 : At equilibrium, for any origin-destination pair (i,j), all the used paths joining i to j have the same cost, and their cost is equal to the cost of the shortest path between i and j.

Note : This theorem corresponds to Wardrop's User Optimal conditions (1952) for the Network Equilibrium Problem.

Proof : Let $k_1 \ldots k_p$ be the used paths (i.e. $h_k > 0$) between a given (i,j) at equilibrium. In (7-1), one must note that the term $S_i(d_i) - I_j(b_j)$ does not depend on k. Since (7-1) must be satisfied for $k_1 \ldots k_p$, then for any k_ℓ, $\ell=1 \ldots p$, $\sum_a \delta_{ak_\ell} C_a(v_a) = -S_i(d_i) + I_j(b_j)$ = constant, and that proves that all the paths $k_1 \ldots k_p$ have the same cost. For the second part of the theorem, let C_{ij} be the cost of the paths $k_1 \ldots k_p$, and suppose that there exists a path k* between i and j satisfying $C_{k*} < C_{ij}$. Since $h_{k*} = 0$, then (7-2) must be satisfied, i.e. $S_i(d_i) + C_{k*} - I_j(b_j) > 0$, or $C_{k*} > - S_i(d_i) + I_j(b_j) = C_{ij}$ which is in contradiction with the hypothesis $C_{k*} < C_{ij}$. Consequently such a k* does not exist, and C_{ij} is then the cost of the shortest path between i and j.

In the following, let K^+_{ij} be the set of used paths between i and j (i.e. having positive flow).

According to theorem 4, at the optimum, between a pair (i,j), all the used paths are shortest paths. This idea suggests, that at each iteration, one generates the shortest paths between the pairs (i,j) and to add these shortest paths to the current set of used paths. When the flow on a path falls to zero, one can remove this path from the set of used paths.

This strategy can be used with any classical algorithm, but, in the following, we will propose the use of the GSN type algorithms described in a previous section. We will call this algorithm GSNPR or GSNrR, where the R component corresponds to this Restriction strategy; restriction to the used variables.

GSNPR Algorithm

(GSNPR) : 0- Initialize $K_{ij}^{+}=\emptyset$, for all (i,j) .

1- For every pair (i,j) DO :

i) Generate the shortest path k between i and j: $K_{ij}^{+}=K_{ij}^{+}\cup\{k\}$

ii) For every $k\in K_{ij}^{+}$ DO :

Solve, by using NP the one variable suproblem in h_k by keeping all other path flows at their current value.

$$\text{Min } \int_0^{h_k+\bar{d}_i} S_i(x)dx + \sum_{a\in k} \int_0^{h_k+\bar{v}_a} C_a(y)dy - \int_0^{h_k+\bar{b}_j} I_j(z)dz \quad (SP_k)$$

subject to $h_k > 0$

where $\bar{d}_i = \sum_{j'\neq j} \sum_{k\in K_{i'j'}} h_k$,

$\bar{b}_j = \sum_{i'\neq i} \sum_{k\in K_{i'j'}} h_k$,

$\bar{v}_a = \sum_{i',j'} \sum_{k\in K_{i'j'}} h_k$ for $i'\neq i$ and $j'\neq j$.

If the optimal solution of (SP_k) is $h_k = 0$ then $K_{ij}^{+} = K_{ij}^{+} - \{k\}$

2- Optimality test : STOP or return to Step 1.

In the solution of (SP_k) by (NP) one needs $g_k(h_k)$ and $g_k'(h_k)$, the first and second derivatives of the objective function with respect to h_k. These are easily shown to be :

$$g_k(h_k) = S_i(h_k+\bar{d}_k) + \sum_{a\in k} C_a(h_k+\bar{v}_a) - I_j(h_k+\bar{b}_j) \qquad (11)$$

and similarly

$$g'_k(h_k) = S'_i(h_k+\bar{d}_k) + \sum_{a\in k} C'_a(h_k+\bar{v}_a) - I'_j(h_k+\bar{b}_j) \qquad (12)$$

We must note that GSNrR can be stated in the same way than GSNPR by replacing in the latest (NP) by (Nr) in step 1-ii-a.

THEOREM 5 : If the objective function f(h) in (8) is convex, low bounded and attains its minimum on {h>0}, then GSNrR generates an optimal solution to (8). If in addition, the functions $S_i+C_k-I_j$ satisfy the conditions of theorem 3, then GSNPR too generates an optimal solution to (8).

Proof : At each iteration, GSNrR adds, for any (i,j), the shortest path k to K^+_{ij} and solves SP^n_q for every q in K^+_{ij}. Then, the objective function cannot do else but decrease. Since the problem is supposed low bounded, then the sequence $\{f(h^n_k)\}$ generated but GSNrR will converge. Suppose now that this sequence does not converge to the optimal value of the problem, then at the limit, at least one of the equilibrium conditions will not be satisfied, i.e. for a (k,i,j), the system (13) is not satisfied:

$$\begin{cases} h(S+C-I)=0 \\ h > 0 \\ S+C-I > 0 \end{cases} \qquad (13)$$

Since h is always > 0 in the algorithm, then only two cases may happen, either h=0 or h>0 :

Case 1 : $h_k>0$ and (13) non satisfied, then $S_i+C_k-I_j\neq 0$, and k is in K^+_{ij}, then at the next iteration, a linear search along k will imply a (strict) decrease of the objective because $h_k>0$ and ($S_i+C_k-I_j\neq 0$), then contracdiction.

Case 2 : $h_k = 0$ and (13) non satisfied, then $S_i+C_k-I_j<0$. Consider a path ℓ such that $S_i+C_\ell-I_j = \min \{S_i+C_q-I_j<0 \ q \text{ in } K_{ij} \text{ and } h_q = 0\}$. Since S and I do not depend on q, then ℓ is the shortest path, having a null flow, between i and j, that do not satisfy system (13). Consider the two possibilities :

2a : $K^+_{ij} = \emptyset$, i.e. all the paths between i and j have a null flow. Then ℓ is the shortest path between i and j, and at the next iteration ℓ is generated, and since $S_i+C_\ell-I_j \leq 0$, a linear search along that path will induce a (strict) decrease of the objective, then contradiction.

2b : $K^+_{ij} \neq \emptyset$, if there exists a path q in K^+_{ij} such that (13) is not satisfied, then we fall in case 1, otherwise $S_i+C_q-I_j = 0$ for all the paths in K^+_{ij} and then $C_\ell \leq C_q$ for all q in K^+_{ij}, and $C_\ell < C_q$ for all q not in K^+_{ij}, then ℓ is the shortest path between i and j and will be generated at the next iteration inducing a (strict) decrease in the objective value because $S_i+C_\ell-I_j< 0$. o

Note : This proof requires that the subproblems SP are solved to optimally in the algorithm, and this is done by (Nr), showing that GSNrR solves the problem. If the subproblems SP are such that one of the three conditions of theorem 3 is satisfied, then GSNPR too solves the problem.

It can be easily shown, under the usual assumptions that the S_i's and C_a's are monotone and I_j's antitone, that f(h) is convex.

The main advantage of the algorithm just described is that it generates itself the variables needed to solve the problem. Hence, there is no need to know a priori all the paths on the network. In addition to this, since the sub-iterations 1-ii are made on K^+_{ij} instead of K_{ij}, the number of subproblems to be solved is less than in a classical GSN type algorithm.

In step 1 of the algorithm, we have a loop on the indexes (i,j), and in the implementation of the algorithm, in our sense, there is no a priori advantageous sequence to be followed, for the choice of pairs (i,j), but the sequence (1,1), (1,2), ..., (2,1), (2,2), (2,3), ... This means that the manner one will number the nodes will influence the efficiency of the algorithm.

To improve the algorithm, we propose to use the augmented network in which there is only one origin-destination pair (m,m). The sets K_{ij} (and K^+_{ij}) are then reduced to only one set K_{mm} say K (and one set K^+_{mm} say K^+). At each iteration, when a shortest path is generated between m and m, a pair (i,j)(through which passes that path) is then automatically selected, and this pair is the best pair in the sense that among all the pairs (u,v), it is that pair (i,j) that holds the shortest path.

Using GSNPR on the augmented network, there is no more necessary to follow a defined sequence on the treatment of O/D pairs, the algorithm will chose itself the best pair to be treated, and the best path of this pair to be treated. In A^+, GSNPR can be stated as follows :

Algorithme GSNPR-A^+ :

(GSNPR-A^+) : 0 - Initiate $K^+ = \emptyset$.

1 - i) Generate the shortest path k between m and m and let $K^+ = K^+ \cup \{k\}$.

ii) FOR every k in K^+ DO :

a) Solve, using (NP), the 1-variable subproblem in h_k by keeping all other path flows at their current value

$$\min \sum_{a \in k} \int_0^{h_k + \bar{v}_a} C_a(y)dy \qquad (SP_k)$$

where $\bar{v}_a = \sum_{\substack{k' \in K^+ \\ k' \neq k}} h_{k'}$.

b) If the optimal solution to $(SP)_k$ is $h_k = 0$
then let $K^+ = K^+ - \{k\}$.

2 - Optimality test : stop or return to 1- .

4. ALGORITHMS'S IMPLEMENTATION

Data structure :

One must note that each variable of the problem corresponds to a path which is a sequence of arcs. Each variable must then be characterized by a vector (or a linked list). Another pointed list must be used to identify and link up the elements of K^+.

Shortest path algorithm :

Dikjstra's algorithm is one of the most efficient algorithms to compute shortest paths between an O/D pair. The only restriction, in using that algorithm, is that all the costs on the network must be non negative. In the original network of the spatial price equilibrium problem, all the transportation costs on the arcs are non negative, and it is possible to use the Dikjstra's algorithm to compute efficiently shortest paths between pairs (i,j). In the case of the augmented network, we have seen that the negative costs are on the (j,m)'s arcs, and only on these arcs.

To solve the problem, we (fictitiously) invert the sense of the arcs (i.e. an arc (u,v) will be seen as (v,u)). At the first step of Dikjstra's algorithm to compute a shortest path from m to m, all the negative arcs will then be used, and the destinations j labeled with an initial cost (which is negative but no matter). In the remainder of the procedure, the negative arcs will never be used again, and the algorithm will continue to work only with positive arcs.

Computation of g and g' :

In the Newton's method, the most consuming time will be the computation of g_k and g'. As it has been shown in (11) to (12), these quantities depend on : $\sum_j \sum_{k \in K_{ij}} h_k = d_i$, $\sum_{i,j} \sum_{k \in K_{ij}} h_k = v_a$, and $\sum_i \sum_{k \in K_{ij}} h_k = b_j$. To improve the computations, instead of computing each time the three sums, one can create the variables v_a, d_i, and b_j's, and each time a h_k is modified, one updates the v_a, d_i, and b_j's corresponding to that h_k.

Termination Criterion (TC) :

In our implementation, we used the following (TC) :

GSNPR-(TC) : for all (i,j) $$\max_{k \in K^+_{ij}} \{|S_i + C_k - I_j|\} < \varepsilon_1 \qquad (14)$$

$$\text{and} \quad |C_{k*(ij)} - |\max\{C_k : k \in K^+_{ij}\}| < \varepsilon_2 \qquad (15)$$

In the case of GSNPR-A^+ where the S_i's and I_j's are integrated in the C_a's, the (TC) becomes :

GSNPR-A^+-(TC) : $$\max_{k \in K^+} \{|C_k|\} < \varepsilon_1 \qquad (16)$$

$$\text{and} \quad |C_{k*} - |\max\{C_k : k \in K^+\}|| < \varepsilon_2 \qquad (17)$$

where ε_1 and ε_2 are tolerance coefficients, that we took to be 1.E-4.

Note that (16) is to satisfy (for ε-optimality) equation (7), and that (17) is to be sure that all used paths have a cost equal to the shortest path's cost (as demanded by theorem 4).

5. NUMERICAL RESULTS

It is known that the Gauss-Seidel type algorithms have usually a linear convergence rate (Fletcher, 1980, p. 102), with regard to the Newton method, its rate of convergence is quadratic (Luenberger, 1973, p. 225-227). As a consequence, the GSN type algorithms have a linear rate of convergence but at each iteration, the subproblems are solved with quadratic convergence's rate. The effect of the component R (restriction) in GSNPR is to reduce the number of subproblems to be treated at each iteration, and this speeds up the computations.

In the following, we will present some numerical tests of the GSNPR algorithm which has been implemented on the Cyber 835 of the Université de Montréal via a FORTRAN code. Table 1 gives results of that program applied on randomly generated networks. The columns "GSNPR" are relative to the original network, and the columns "GSNPR-A^+" to the same problems seen on the augmented network. In the column "MINOS", the same problems are solved by the program MINOS (Murtagh and Saunders, 1977), but in this case, we have used the problem formulated in the arc flow variables space, problem which is a more easier problem than the one formulated in the path flow variables space. We did not apply MINOS to the latest problem because the lonely computation time to generate the paths has been greater than the GSNPR computation time to solve the problem, and this for the smallest problem in table 1. In addition to this, each path is a sequence of arcs. All this is to say that a classical algorithm is absolutely not adequate to solve the Spatial Price Equilibrium Problem formulated in the path flow variables space.

Table 1

Size	T	P	GSNPR			GSNPR-A^+			MINOS
			PM	NNEWT	s.CPU	PM	NNEWT	s.CPU	s.CPU
10	C		22	1700	7,00	11	1025	1,81	2,16
/	L	1597	26	542	2,44	13	522	1,52	2,27
32	Q		27	812	3,52	16	650	1,23	2,48
20	C	>				23	12094	30,82	6,99
/	L	10000			> 190	41	3531	10,31	15,35
117	Q					46	4507	12,36	16,20
30	C	>				39	20934	73,80	18,30
/	L	10000			> 190	63	8238	37,10	39,90
251	Q							> 190	51,50

where :

Size : number of nodes / number of arcs of the network ;

T : indicates the type of the transportation cost functions on the arcs
C = constant
L = linear
Q = quadratic ;

P : approximative number of paths (variables) of the problem ;

PM : mean number of paths used at each iteration by GSNPR (GSNPR-A^+) ;

NNEWT : number of calls to the Newton subroutine to solve a 1-variable subproblem SP ;

s.CPU : CPU time used to solve the problem on the Cyber 835 of the Université de Montréal, (GSNRP, GSNPR-A^+, MINOS) .

The columns PM in GSNPR and GSNPR-A^+ show how much lower is the (mean) number of paths, used in these algorithms, than the number of paths (in P) that would have been used by a classical algorithm. The columns s.CPU in GSNPR and GSNPR-A^+ show that seeing the problem in the augmented network improves very much the algorithm GSNPR. We can also note that in spite of solving an easier problem (in arc flow variables) by MINOS, this latest is not much more efficient than GSNPR-A^+ (at least for problems in the size of those in table 1).

To end this discussion, we would like to attract the reader's attention to the fact that all the problems in table 1 have a non linear objective even if in problems C and L the cost functions are constant or linear. In fact, one part of the objective depends on the offer and demand functions which are taken to be non linear in all these problems, and the other part, which depends on the arcs' cost functions, is of the form $\int C_a(x)dx$ and is linear for the lonely case where the C_a's are constants.

Conclusion

To solve the Spatial Price Equilibrium Problem formulated in terms of path flow variables, this paper proposes a Gauss-Seidel-Newton-Projection with Restriction to non negative variables. This restriction is obtained by generating, at each iteration, the shortest paths from all the origins to all destinations, and by adding them to the actual set of used paths.

The proposed algorithm does not request the explicitation of all the variables a priori; the variables (paths) needed to solve the problem are generated during the execution by the shortest path procedure. On the other side, this restriction permits to reduce the number of variables used per iteration. In our numerical tests, this reduction has been of the order of 1/1000 for some problems, and this rate of 1/1000 is not a lower bound.

We showed that the proposed algorithm (GSNPR) may be improved if the Spatial Price Equilibrium Problem is seen as Network Equilibrium Problem (i.e. in an augmented network A^+), and the numerical tests granted that proposition.

The algorithms GSNPR and GSNPR-A^+ have been implemented on the CYBER 835 of the Université de Montréal via a FORTRAN code. The GSNPR-A^+ algorithm has been very efficient to solve problems with more than a 1000 paths, and its computation time was in the same order of the computation

time used by the program MINOS to solve the same problems formulated in the arc flow space (which is an easier problem).

In the future, some Newton methods will be investigated for the Spatial Price Equilibrium Problem formulated in terms of arc flow variables. The problem with a symetric Jacobian will be first treated, and we will then have a look to the problem with a non symetric Jacobian, problem that cannot be formulated as a mathematical program, but as a variational inequality.

BIBLIOGRAPHY

AHN, B.H. and SEONG, B.S. (1984), "A Parametric Method for Computing Non Linear Spatial Equilibrium", Department of Management Science, Korea Advanced Institute of Science and Technology.

FLETCHER, M. (1980), Practical Methods of Optimization, Vol. I, John Wiley and Sons, 77-99, 102.

FLORIAN, M. (1981), "On some Generalizations of Transportation Problems", Operationsl Research '81, P. Brans (Ed.), IFORS, 607-620.

FLORIAN, M. and LOS, M. (1982), "A New Look at Static Spatial Price Equilibrium Models", Regional Science and Urban Economics 12, 579-597.

GUDER, F. and MORRIS, J.G. (1985), "A Two Line Algorithm for Computing Equilibrium Single Commodity Trade Flows", Technical report, Loyola University of Pennsylvania.

LUENBERGER, D.G. (1973), Linear and Non Linear Programming, Chap. 6, Addison-Wesley.

MacKINNON, J.G. (1975), "An Algorithm for the Generalized Transportation Problem", Regional Science and Urban Economics 5, 445-464.

MURTAGH, B.A., and SAUNDERS, M.A. (1977), "MINOS User's Guide", Technical report 77-9, Systems Optimization Laboratory, Stanford University, Stanford, CA.

NAGURNEY, A. (1985), "An Algorithm for Single Commodity Spatial Price Equilibrium Problem", Technical report, School of Management Science, University of Massachusetts.

NAGURNEY, A. (1985), "Computational Comparisons of Spatial Price Equilibrium Methods", Technical report, School of Management Science, University of Massachusetts.

ORTEGA, J.M. and RHEINBOLDT, W.C. (1970), Iterative Solution of Non Linear Equations in Several Variables, Chap. 13, Academic Press.

PANG, J.S. (1981), "A Hybrid Method for the Solution of Multi-Commodity Spatial Price Equilibrium Problems", Management Science 27.

SAMUELSON, P.A. (1952), "Spatial Price Equilibrium and Linear Programming", American Economics Review 42, 283-303.

TAKAYAMA, T., and JUDGE, G.C. (1971), Spatial and Temporal Price Allocation Models, North-Holland.

TOBIN, R.L. and FRIESZ, T.L. (1983), "Formulating and Solving the Spatial Price Equilibrium Problem with Transshipment in Terms of Arc Variables", Journal of Regional Science 23(2), 187-198.

WARDROP, J.G. (1952), "Some Theoretical Aspects of Road Traffic Research", Proceedings of the Institute of Civil Engineers, Part II, 325-378.